Mean Square Displacement (MSD) calculation

The definition[1-4]

A single-particle trajectory consists of a sequence of N particle positions $\{r_i\}_{i=1}^{N} = \{x_i, y_i, z_i\}_{i=1}^{N}$ observed at specific times $\{t_i\}_{i=1}^{N}$ separated by time step *dt*. The mean square displacement (MSD) is then computed for time intervals (/lags) τ according to

$$MSD(\tau) \equiv \left\langle \Delta r^2(\tau) \right\rangle$$

τ, time interval (/lag); △r, distance …(1)

$$= \left\langle \left| r_{i+\tau} - r_i \right|^2 \right\rangle$$

i, 1~N positions; τ, time interval (/lag) …(2)

$$= \frac{\sum_{i=1}^{N-\tau} \left(\left\langle (x_{i+\tau} - x_i)^2 + (y_{i+\tau} - y_i)^2 \right\rangle \right)}{N - \tau}$$

…(3)

N, number of positions
i, 1~N positions
τ, time interval (/lag)
x, *y*, coordinates of positions

The mean square displacement (MSD) by definition is the (vector) distance (r) traveled by a molecule over some time intervals (/lags) of length τ (**equation 1**).

The squared distance (/displacement) from position *i* to position *i* + τ is averaged over many such time intervals (lags) τ (**equation 2**).

Precisely, at each time interval (/lag) τ, for a single trajectory of N steps, there are N - τ number of sub-trajectories. Average of these time-interval-wise sub-trajectories summarizes behavior of a trajectory into one number at this time interval (/lag) τ (**equation 3**).

Often this quantity is averaged also over all molecules in the system, produce a summarized MSD for behavior of all molecules measured at specific time interval (/lag) τ in the system.

A diagram

The mean square displacement (MSD) definition graph,

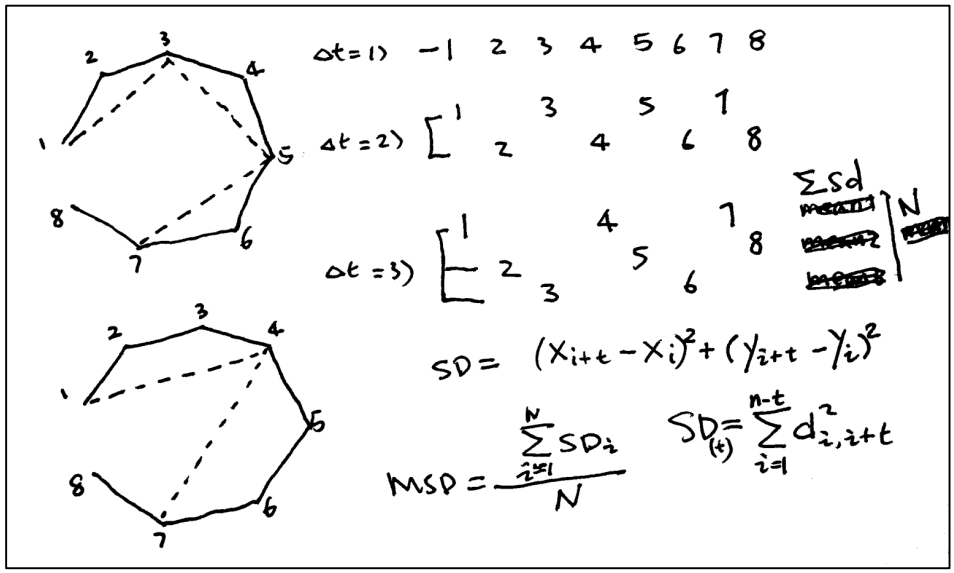


Figure 1. Illustration of sampling /segmentation of trajectory for time lags (τ) of 1, 2 and 3 time steps (*dt*). dt and sub-trajecotry is listed.

The mean square displacement (MSD) is a measure of the average distance a molecule travels. It is the averaged displacements of a particle measured at incremental time intervals. Each time interval (*dt*) measured gives a MSD value of the trajectory (Fig. 1 and 2).

From MSD, the type of motion that a molecule follows can be obtained by plotting it against τ, the number of time intervals analyzed [5] (Fig. 2). The MSD plot is time interval (*dt*) dependent but is not restricted by (/independent of) specific time units (e.g. *ms*).

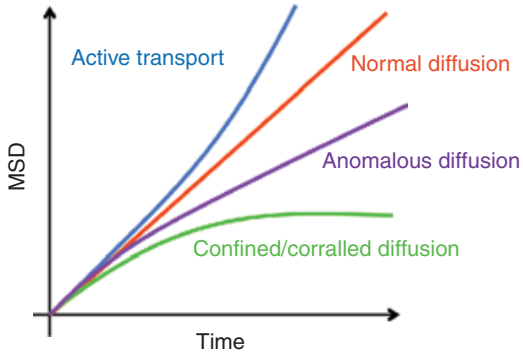


Figure 2. Type of diffusion behavior can be inferred from MSD plot.

msd() function algorithm[1]

msd() calculate track (/trajectory)'s mean square displacement as a function of time (dt). For a single trajectory of N steps, at each dt, there are **N-dt** number of sub-trajectories (/subtracks), average of dt-wise sub-trajectories summarizes sub-trajectories into one number at each dt.

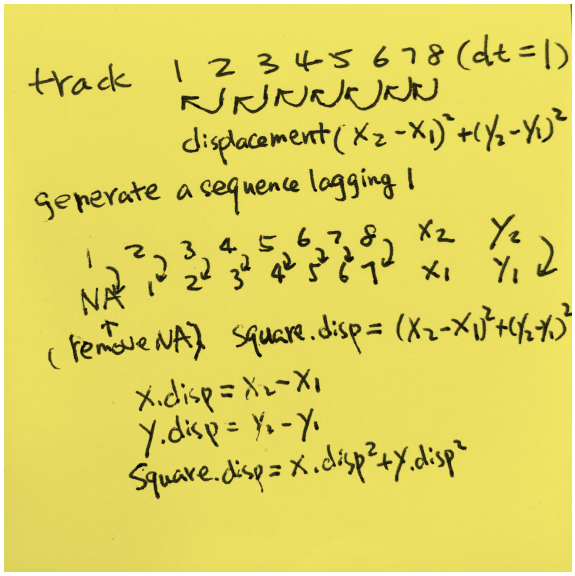
square.disp (track, dt)

track, a data.frame that has two column, x and y.
dt, number of time intervals

This function calculates squared displacement of a particle, for a specified time lags (τ) of 1, 2, 3, … ,τ time steps (*dt*). It is the workhorse of msd calculation.
(TODO: use c++ to speed up this program as it has been called millions of times)

Diagrammatic and descriptive description of algorithm

[1] This documentation uses descriptive algorithm (in words) and a descriptive diagram (in picture) to explain the algorithm of major functions.

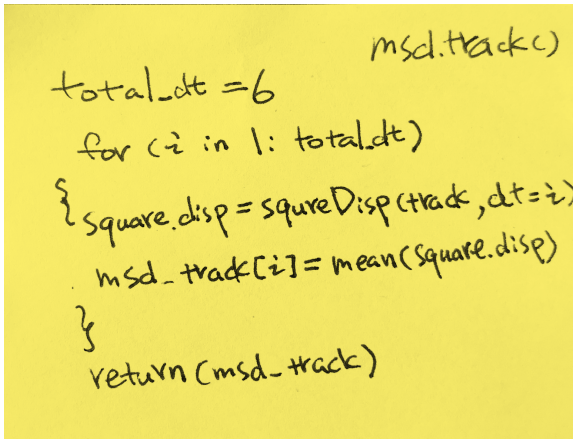


The key algorithm is to generate a lagged sequence to compute x, y displacement. A new sequence lagging 1 in position, then use old position and the new position to compute $x_{i+\tau} - x_i$ as well as $y_{i+\tau} - y_i$ at specified τ for a trajectory.

Note:
the parameter dt in squareDisp() and msd.track() has different meaning. in squareDisp() it means "AT specific number (τ) of time intervals"; while in msd.track(), it means /calculates "1~τ number of time intervals". (TODO: rename the msd.track one to avoid confusion)

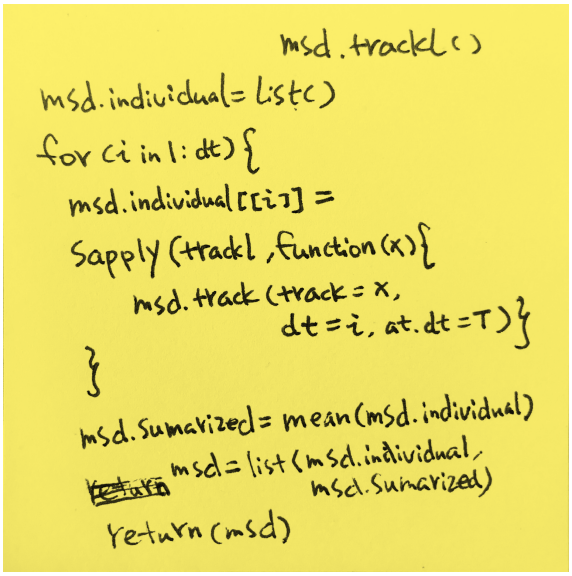
msd.track(track, dt=6)

This is the definition of MSD for a single track. It calculates squared displacement for a trajectory from one to a specified time interval /lags (τ), and then output its mean. It calls function square.disp to calculate squared displacement at individual time lags.



(TODO: need to replace dt with increment to avoid confusion. This dt is more the definition of τ.)
(TODO: resolution variable in original function should be defined as a global constant in the environment. At current stage, make the function more stable, as global variable is not everywhere, for the sake of parallel)

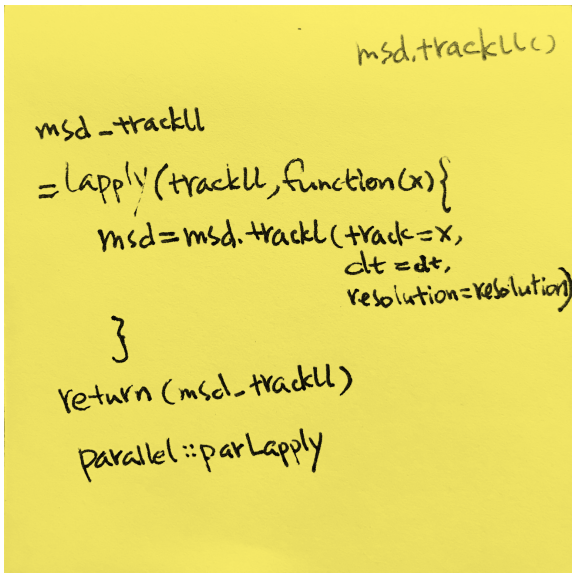
msd.trackl(trackl,dt=6,resolution=0.107)



```
                    msd.trackl()
msd.individual= list()
for (i in 1:dt){
  msd.individual[[i]] =
  sapply (trackl, function(x){
      msd.track (track = x,
                 dt = i, at.dt = T)}
  }
  msd.Sumarized = mean(msd.individual)
  msd = list(msd.individual,
             msd.Sumarized)
  return (msd)
```

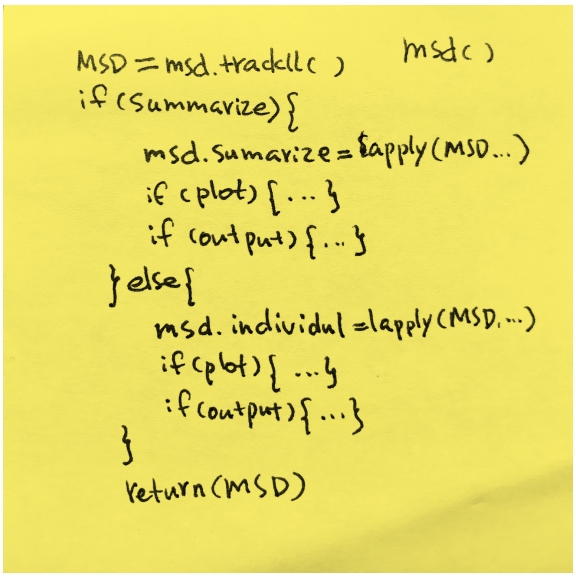
calculates msd.track for a list of tracks, trackl. It applies msd.track to a list of tracks. It outputs individual and summarized MSD, standard deviation and number of tracks at each dt specified.

msd.trackll(trackll,dt=6,resolution=0.107,cores=1)



This function applies msd.trackl() to a folder of list, trackll. If cores are more than one, it assigns lists to CPU cores to compute msd.trackl() in parallel.

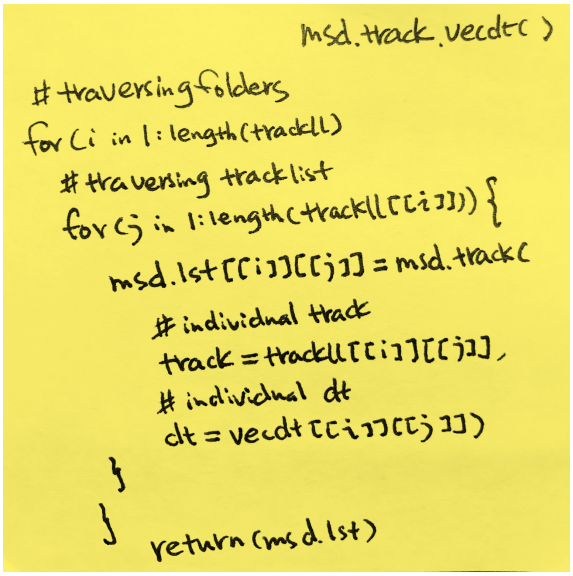
msd()



This is a wrapper /user interface of msd.trackll() function. It computes msd for single trajectory as well as summarized msd for all trajectories over a specified time lags (τ) of 1, 2, 3, … ,τ time steps (*dt*) in the system. It calls msd.trackll() function (can be paralleled), and does plotting and output msd csv files.

functions outside trackll includes output summarized MSD and individual MSD as matrix and plot each correspondingly.

msd.track.vecdt()



```
msd.track.vecdt( )

# traversing folders
for (i in 1:length(trackll)
  # traversing tracklist
  for (j in 1:length(trackll[[i]])) {
    msd.lst[[i]][[j]] = msd.track(
      # individual track
      track = trackll[[i]][[j]],
      # individual dt
      dt = vecdt[[i]][[j]])
  }
}
  return (msd.lst)
```

This function is a special case of msd.track(), where dt is of different values, corresponding to the first 25% of the total length, when calculate MSD for each track.

It uses for loop i, j variable to traverse through folders and tracklists, subsetting individual tracks, as well as individual dt, for msd.track() to compute its corresponding msd.

It is specifically designed for diffusion coefficient calculation using percentage method (see Dcoef.perc for details).

Diffusion coefficient (Dcoef) calculation

The definition

Molecules exhibit Brownian motion follow *Einstein-Smoluchowsky* equation,

$$MSD(\tau) \equiv \left\langle \Delta r^2(\tau) \right\rangle = \left\langle \left| r_{i+\tau} - r_i \right|^2 \right\rangle = 2dD\tau \qquad \ldots(4)$$

τ, time interval (/lag); $\triangle$r, distance; i, 1~N positions;
d, number of dimensions of data;
D, self-diffusion coefficient

The limiting slope of $MSD(\tau)$ at dimension d, is close to the self-diffusion coefficient D, multiplied by $2d$.

$$\lim_{\tau \to \infty} \frac{d}{\tau} \left\langle \left| r_{i+\tau} - r_i \right|^2 \right\rangle = 2dD \qquad \ldots(5)$$

τ, time interval (/lag); $\triangle$r, distance; i, 1~N positions;
d, number of dimensions of data;
D, self-diffusion coefficient

As τ gets closer to infinity ∞, on the function $MSD(\tau)*d/\tau$, *i.e.* the slope of MSD at dimension d, its value gets closer to $2dD$.

Therefore, self-diffusion coefficient D can be calculated /approximated by linear fit $MSD(\tau)$ curves between several time lags to get the slope and then divide it by $2d$ (Figure 1).

A diagram

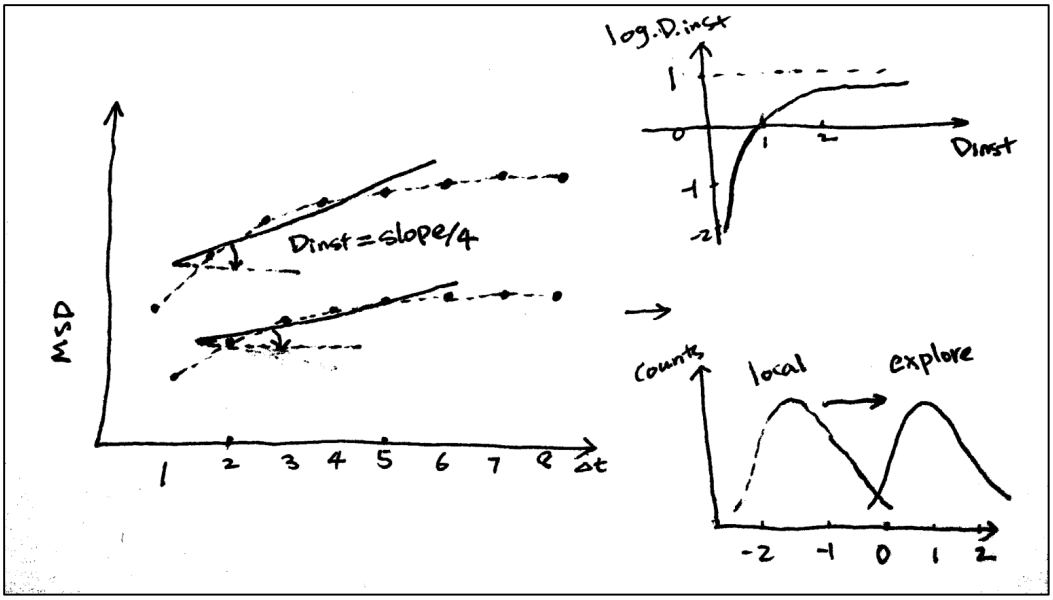


Figure 1. Diagram of diffusion coefficient calculation. Diffusion coefficient is calculated as the slope of the linear fit between time lags 2~5, then divided by 2*d*=4. The derived coefficient is usually log transformed to better differentiate populations (Figure1 C). The log transformation better differentiates D (-∞, 1) and D (1, ∞) separated by free diffusing D=1, log10(1)=0 (Figure1B).

Ideally, the number of time lags (τ) considered for fitting should be sufficiently large for the limiting slope of $MSD(\tau)$ to be in the linear regime for accurate fitting[2,3] (*i.e.* the $MSD(\tau)$ plot appears more linear for better fitting of slope function) (Figure 5 of Qian et al 1991). However, diffusion coefficient calculated from least square fitting on more than first 8 time lags are heavily influenced by noise in approximating true coefficient [4](Figure 2 of Saxton et al 1997).

For this reason, part of trajectory that extends over 32 frames is considered non-informative in approximating diffusion coefficient and is therefore trimmed off; for the remaining trajectory and trajectories that has length less than 32 frames, initial 25% time lags of the MSD curve are used for fitting[6], this ensures less than 8 time lags are used for fitting to approximate diffusion coefficient.

Because the trajectory's length (N) varys, the number of time lags used for fitting (N/4) can also vary between trajectories. To reduce this variation, this package implemented

a tiered percentage to give a relatively consistent number of time lags (*dt*) for each trajectory despite the changes in their length (see t able 1).

Table 1. Tiered percentage method for diffusion coefficient calculation

Trajectory Length (# frames)		**Percentage**	**Time lags (dt) excluded**			**Time lags (dt) used for fitting**			
			Initial time lag	Last time lag					
Low	High			Low	High	Low	High	Low	High
31	/	0.25	1	6	8	2	5	2	7
22	30	0.25	1	6	8	2	5	2	7
15	21	0.4	1	6	8	2	5	2	7
10	14	0.6	1	6	8	2	5	2	7
7*	9	1	1	6	8	2	5	2	7
5**	6	1	n.a.	n.a.	n.a.	1	4	1	5

* 7 frames give maximum 6 time lags (dt). It is the lower bind for using consistent amount of time lags for fitting, corresponding to different trajectory length. As a reference, the static method for diffusion coefficient calculation is using time lags (dt) from 2~5 for all trajectories despite their length.

** For trajectories that has less than 7 frames, using all time lags for fitting. By default, the minimum trajectory length is 5 frames, which gives 4 time lags (dt) for fitting; there is no limit on maximum trajectory length, however only the first 31 frames are considered, which yield maximum 8 frames (7 time lags) for fitting with percentage default at ¼.

An alternative approach is to stabilize the number of time lags used for fitting using only time lag 2 and 5 despite the total time lags measured {Normanno:1fl} (table 2). This method has the advantage of consistent measuring initial stage of the diffusion, therefore also termed “instantaneous” diffusion coefficient. Due to its simplicity, it’s been adopted the default method in this package.

Table 2. Static method for diffusion coefficient calculation

Trajectory length		**Time lags used for fitting**	
low	high	low	high
7	/	2	5

To be continued..

Dcoef() function algorithm

Dcoef.static()

This function implements the static method for diffusion coefficient calculation.

Dcoef.perc()

Dcoef()

msd.perc()

CDF based diffusion coefficient[7]

Reference

1. Monnier, N. *et al.* Bayesian Approach to MSD-Based Analysis of Particle Motion in Live Cells. *Biophysical Journal* **103,** 616–626 (2012).
2. Wiketomica. Lennard Jones Mean Square Displacement. *rheneas.eng.buffalo.edu* Available at: http://rheneas.eng.buffalo.edu/wiki/LennardJones:Background:Lennard_Jones_Mean_Square_Displacement. (Accessed: 9 November 2016)
3. Qian, H., Sheetz, M. P. & Elson, E. L. Single particle tracking. Analysis of diffusion and flow in two-dimensional systems. *Biophysj* **60,** 910–921 (1991).
4. Saxton, M. J. Single-particle tracking: the distribution of diffusion coefficients. *Biophysj* **72,** 1744–1753 (1997).
5. Ruthardt, N., Lamb, D. C. & Bräuchle, C. Single-particle Tracking as a Quantitative Microscopy-based Approach to Unravel Cell Entry Mechanisms of Viruses and Pharmaceutical Nanoparticles. *Mol Ther* **19,** 1199–1211 (2011).
6. Tarantino, N. *et al.* TNF and IL-1 exhibit distinct ubiquitin requirements for inducing NEMO–IKK supramolecular structures. *J Cell Biol* **204,** 231–245 (2014).
7. Vrljic, M., Nishimura, S. Y., Brasselet, S., Moerner, W. E. & McConnell, H. M. Translational diffusion of individual class II MHC membrane proteins in cells. *Biophysical Journal* **83,** 2681–2692 (2002).